REMARKS

Claim rejections under 35 USC 112

Claims 8, 12, and 16 have been rejected under 35 USC 112, as failing to set forth the subject matter that Applicant regards as his invention. With respect to claim 8, the phrase "more simply" has been deemed to be indefinite. Without prejudice, Applicant has amended claim 8 to remove this phrase, and therefore requests the withdrawal of this rejection as to this claim.

With respect to claims 12 and 16, the phrase "seamlessly intermix" has been deemed to be indefinite. Applicant cannot find this phrase in claim 12, and therefore submits that claim 12 is definite as currently written. With respect to claim 16, claim 16 has been cancelled without prejudice. Therefore, Applicant requests the withdrawal of this rejection as to claims 12 and 16.

Claim rejections under 35 USC 101

Claims 14-17 and 18-20 have been rejected under 35 USC 101 as being directed to non-statutory subject matter. With respect to claim 14, the Examiner has stated that claim 14 fails to result in a physical transformation or produce a concrete tangible result. Applicant has amended claims 14 and 18 so that the AST is stored, which results in a physical transformation of the storage medium on which the AST is stored. As such, claims 14 and 18 as amended are statutory under 35 USC 101, and Applicant requests the withdrawal of this rejection as to claims 14-17 and 18-20 where claims 15-17 and 19-20 ultimately depend from claim 14 or claim 18.

Claim rejections under 35 USC 102

Claims 1-20 have been rejected under 35 USC 102(e) as being anticipated by Morshed (6,760,903). Claims 1, 14, and 18 are independent claims, from which the remaining pending claims ultimately depend. Applicant submits that as originally presented or as has been amended, each of claims 1, 14, and 18 is patentable over Morshed, such that all the claims rejected on this basis are patentable.

Independent claims 1, 14, and 18 are each limited to a probe program that is executable by or via an interpreter. The probe program is further independent of the architecture of processors (i.e., it is hardware independent). Applicant submits that these claim limitations are not disclosed, suggested, or taught by or within Morshed.

It appears that the Examiner has equated the various monitoring libraries (i.e., dynamically linked libraries, or DLL's) of Morshed as corresponding to the probe program of the claimed invention. For instance, the Examiner states that "[t]he library (probe program associated with each breakpoint) may include code that is invoked to gather various types of performance information." (Office action, p. 4) If Applicant is incorrect in this conclusion, the Examiner is requested to identify what elements of Morshed the Examiner is relying upon as corresponding to the probe program of the claimed invention. In this respect, Applicant notes that 37 CFR 1.106(b) states that "[w]hen a reference is complex or shows or describes inventions other than that claimed by the applicant, the particular part relied on must be designated *as nearly as practicable.*" (Emphasis added) Morshed is most definitely a complex reference; indeed, it numbers 121 pages! Therefore, the Examiner is required to indicate which portion of Morshed the Examiner is relying upon as corresponding to the probe program of the claimed invention.

Applicant submits that the monitoring libraries, or DLL's, of Morshed (i.e., the probe programs of Morshed) are not particularly described in Morshed as being (1) executable via an interpreter; and (2) independent of the architecture of the processors, as to which the claimed invention is limited, and as is now particularly discussed in detail.

Probe program executable via an interpreter

First, the Examiner has referenced column 41, line 43 of Morshed as indicating that the probe program of Morshed is executable by an interpreter. However, column 41, line 43 only states that "Java code . . . may be interpreted using the JavaVM" (i.e., virtual machine). It does not say that the probe program itself (i.e., the monitoring DLL) is interpreted.

The relied upon excerpt of Morshed in relevant part specifically states the following:

> The techniques described . . . may be used, for example, with Java code as may be interpreted using the JavaVM. . . . . The client browser receives the page, *begins executing Java code* that may be included in the returned web page [where presumably this Java code is executed by an interpreter] *resulting in control being passed to the JavaVM DLL. The instrumented JavaVM byte code is executed and communicates with the monitor DLL* . . . . In one embodiment, control may be initially transferred to the monitor DLL.

(Col. 41, ll. 41-62) Note what is going on here. The Java VM can interpret Java code. However, the Java VM is described as communicating with the monitoring DLL, and indeed, initially transferring control to the monitoring DLL. Thus, the monitoring DLL – i.e., the probe program of Morshed – is not actually described as being interpreted by an interpreter. The Java VM DLL interprets Java code, and can communicate with the monitoring DLL, and in fact can pass control to the monitoring DLL. In the latter instance especially, there is no way the Java VM DLL interprets the monitoring DLL, since it has passed control to the monitoring DLL.

Applicant respectfully submits that the Examiner is misconstruing what a dynamically linked library, or DLL, is. A DLL is a library of executable (not interpreted) code that a given computer program, such as one written in Java interpreted code, is linked to, so that, for instance, the program can run on a given platform. For example, the web site http://www.techweb.com/encyclopedia/printArticlePage.jhtml?term=DLL defines DLL as an "executable program module . . . that performs one or more functions at runtime." DLL's are not interpreted; they are executed. Therefore, a DLL by definition is not executable via an interpreter, such as a Java VM as in Morshed.

In any case, Applicant respectfully submits that the Examiner has not engaged in proper analysis of Morshed in anticipating the claimed invention. To anticipate a claim, the prior art reference must disclose each element of the claimed invention "arranged as in the claim" in question. (Lindermann Maschinenfabrik GmbH v. American Hoist & Derrick Co., 221 USPQ 481, 485 (Fed. Cir. 1984)) In the present situation, the Examiner found in Morshed a probe program – the monitoring DLL – and also where code is interpreted by an interpreter instead of being directly executed. However, in the claimed invention, it is the probe program that is interpreted by an interpreter instead of being directly executed. This *arrangement* of the probe program being interpreted by an interpreter is not disclosed in Morshed. Rather, Morshed discloses a probe program, and does disclose interpretation of code by an interpreter, but does not specifically disclose the probe program itself being interpreted by an interpreter.

Probe program independent of the architecture of the processors

Second, the Examiner has referenced column 19, line 40 of Morshed as indicating that its probe program is independent of the architecture of the processors. However, column 19, line 40 simply states that "[b]yte code is a format that is usually independent of the source language from which it was compiled, and which is intended to be independent of any computer hardware and/or operating system on which it might run." (Col. 19, ll. 39-43) Morshed goes on to say that "[b]yte code programs are executed by a software program sometimes referred to as a virtual machine." (Col. 19, ll. 43-45)

In other words, column 19, lines 39-45 of Morshed do not say that the *probe program* itself is byte code that is architecture independent, as is the probe program of the claimed invention. Rather, Morshed's discussion here just says what byte code is. The fact that byte code is independent of hardware, such as processors, does not mean that Morshed's probe program – its monitoring DLL – is byte code. Again, as stated above, the Federal Circuit in Lindermann has stated that a prior art reference must disclose each element of the claimed invention "arranged as

in the claim." In the claimed invention, it is the probe program that is independent of the architecture of the processors. However, Morshed does not disclose this. It discloses a probe program, and discloses byte code that is architecture independent. Morshed does not disclose, in other words, the probe program itself being byte code that is architecture independent.

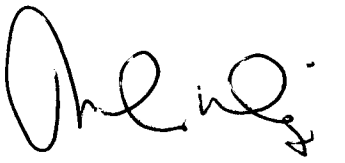
Concluding remarks as to anticipation of the claimed invention

For both of these reasons, Morshed does not anticipate the claimed invention. The Examiner is required to show where in Morshed its probe program – i.e., its monitoring DLL – is interpreted by an interpreter, and where its probe program is independent of the architecture of the processors. In both instances, Morshed discloses interpretation by an interpreter and architecture independence, but *not with specific respect to its probe program*. Furthermore, in the former instance, the DLL that does the interpreting – the Java VM DLL – interprets code and passes information *to the monitoring DLL* and indeed execution control can itself be passed to the monitoring DLL, which suggests that the monitoring DLL is not in fact interpreted by the interpreter (i.e., the Java VM DLL). As such, the claimed invention is patentable over Morshed.

Conclusion

Applicants have made a diligent effort to place the pending claims in condition for allowance, and request that they so be allowed. However, should there remain unresolved issues that require adverse action, it is respectfully requested that the Examiner telephone Mike Dryja, Applicants' Attorney, at 425-427-5094, so that such issues may be resolved as expeditiously as possible. For these reasons, this application is now considered to be in condition for allowance and such action is earnestly solicited.

Respectfully Submitted,



February 26, 2007
Date

Michael A. Dryja, Reg. No. 39,662
Attorney/Agent for Applicant(s)

Law Offices of Michael Dryja
704 228th Ave NE #694
Sammamish, WA 98074
tel: 425-427-5094
fax: 206-374-2819